IS IT WORTH STORING HISTORICAL GRADIENTS?

Yingxin Liu, Joong Ho Choi, Yash Maurya
Carnegie Mellon University
{yingxin2, joonghoc, ymaurya}@andrew.cmu.edu

ABSTRACT

Federated Learning (FL) is a robust machine learning framework allowing multiple entities to collaboratively train a global model without exposing their sensitive data. However, FL's inherent design makes it susceptible to adversarial attacks, making detection and mitigation crucial for the model's integrity and reliability. Traditionally, it was thought that tracking historical weight gradients during training could establish baseline behavior and identify anomalies suggestive of adversarial intrusions. Yet, our research challenges this belief, suggesting that focusing on current weight data is more effective for detecting attacks, such as label flip attacks. This method not only boosts detection accuracy but also enhances storage efficiency, as it avoids the need to store extensive weight gradient histories. Moreover, this approach to data minimization not only augments privacy by concentrating on key data but also lowers the risk of exposing client data, which is in line with privacy-focused machine learning methodologies. GitHub Project: FL

1 INTRODUCTION

Federated learning (FL) provides a structure for training a machine learning model in a decentralized manner, ensuring the privacy of participants. However, the decentralized nature introduces a vulnerability: malicious clients may compromise the global model by sending manipulated local gradients. This risk necessitates the development of robust mechanisms to filter and validate incoming updates, ensuring the security and integrity of FL models.

Recent methods Gupta et al. (2022); Jebreel & Domingo-Ferrer (2022) store historical gradients for identifying the attackers. However, collecting gradients from all clients raises privacy concerns. One way to ameliorate this problem involves integrating differential privacy Dwork (2006) and data minimization techniques at the server level. Differential privacy mechanisms, like injecting noise during model aggregation, protect individual contributions; data minimization at the server involves storing and processing the minimum necessary information, curbing potential privacy breaches. Thus, our study aims to highlight the limited effectiveness of integrating long HoG (history of gradients) into the detection mechanism. By omitting the generation of un-targeted attackers, our focus is exclusively on distinguishing targeted attackers (adversarial clients engaged in label poisoning) from normal clients, utilizing the current gradients of weights. This methodology underscores the marginal impact of historical gradients on both the accuracy of the detection mechanism and the efficiency of server storage, while concurrently enhancing privacy considerations.

2 RELATED WORK

In recent years, there has been a rising enthusiasm for FL, marked by a growing number of applications spanning various fields Li et al. (2020), Niknam et al. (2020), Huang et al. (2019). As the deployment of FL continues to expand, the significance of ensuring FL security and addressing adversarial attacks has become increasingly apparent Kairouz et al. (2021). Within the context of FL, numerous attack strategies have been put forth, encompassing poisoning attacks Tolpegin et al. (2020), Fang et al. (2020), Bhagoji et al. (2019), Shejwalkar & Houmansadr (2021), membership inference attacks Nasr et al. (2019), Truex et al. (2019), backdoor attacks Bagdasaryan et al. (2020), leakage attacks Wang et al. (2018) Mo et al. (2021), and various other approaches Mothukuri et al. (2021).

The stochastic gradient descent (SGD) algorithm has been found to be susceptible to untargeted (Byzantine) attacks, in which malevolent clients send random or arbitrary gradients to the server in

an attempt to impede the convergence or the global model's performance. There have been several prior works Erbil & Gursoy (2022); Jebreel & Domingo-Ferrer (2022) aimed at detecting targeted attackers where they focus on one specific type of attack like targeted data poisoning attacks and

In a recent advancement in this domain, the authors introduced MUD-HoG Gupta et al. (2022), employing short HoG (History of Gradients) for detecting untargeted attacks and unreliable clients, while utilizing long HoG for targeted attack detection. Our empirical findings challenge the necessity of a history of gradients for identifying targeted attacks, demonstrating that relying on current gradients not only enhances overall accuracy but also improves storage efficiency.

3 METHOD

3.1 HISTORICAL GRADIENT FOR ATTACKER DETECTION

Intuitively, historical gradients of clients indicate the direction of model updates, revealing potential anomalies like label poisoning or backdoor attacks. Thus, we hypothesis that by tracking these gradients as trajectories, we can distinguish between malicious and benign clients. Specifically, an attacker's gradient trajectory is expected to diverge from that of benign clients. Moreover, in heterogeneous FL, this divergence becomes to have from the round of the attack happens for the attacker's trajectory, while clients with disparate data distributions show deviation from the start of training. Analyzing these divergences helps identify attackers and non-IID clients, as demonstrated in Fig. 1.

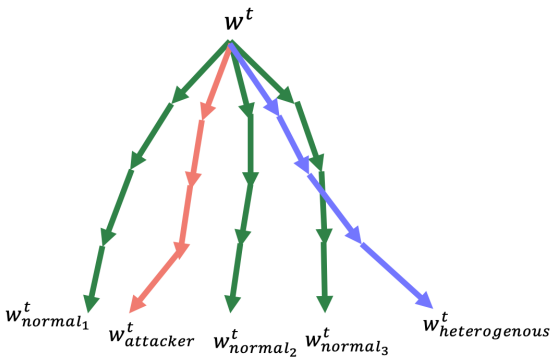

Figure 1: The green gradients of normal clients always perform normally, while the red gradients of attackers diverge from the middle of training, and the purple line of the client with the disparate data distribution diverge from the beginning of training.

3.1.1 MODELING METHOD

To model trajectories, we extract neuron-level statistics as features, where i is the layer index, j the neuron index, and k the client ID. The statistics for client k's neuron j in layer i are $s(i, j, k)$, comprising moments and IQR:

$$s(i, j, k) = [1^{\text{st}} \text{ moment}, 2^{\text{nd}} \text{ moment}, 3^{\text{rd}} \text{ moment}, 4^{\text{th}} \text{ moment}, \text{IQR}].$$

In order to perform classification with these features, we have designed two methods. The first method utilizes K-means clustering, which is straightforward but requires pre-identification of the target number of clusters before classification. To address this limitation, we propose the second method. In this approach, for each feature of each client's trajectory, we calculate its z-score. The z-score for feature x in $s(i, j, k)$ is calculated with mean μ and standard deviation σ of the feature:

$$z_{i,j,k,x} = \frac{s(i, j, k, x) - \mu_x}{\sigma_x}.$$

Subsequently, for each trajectory, we aggregate their z-scores in a weighted manner. The weights are assigned based on the perceived importance of each feature, where $z_{i,j,k} = \sum_{x \in X} w_x z_{i,j,k,x}$.

For instance, if we consider 1^{st} moment to have a greater influence on the trajectory, it is assigned a higher weight w_x. Finally, we compare the sum of each client's weighted z-scores to a predetermined threshold to determine whether it is indicative of an attacker.

3.1.2 ANALYSIS

Unfortunately, our model's performance in classifying based on historical or one-round gradients is not satisfactory, resembling random generation. This is likely because gradients are high-dimensional; in such spaces, data points are widely dispersed, making distance and similarity less intuitive. With vectors scattered across various dimensions independently, modeling historical gradients as a representative trajectory becomes challenging.

3.2 WEIGHT

Following the unsuccessful attempt to classify using gradients, we shifted our focus to employing weights instead. During our experiments, we observed that Gupta et al. (2022) also implemented a classification strategy for identifying attackers based on weights. To be noticed, although Gupta et al. (2022) refers to these as gradients, they essentially utilize weights. In Gupta et al. (2022), the approach involves extracting the mean of historical weights of each client, and then applying K-means classification to these statistics.

However, Gupta et al. (2022) does not provide a comparative analysis between the use of historical weights and one-round weights. Moreover, their experiments do not fully address the challenges in heterogeneous FL environments. Although they simulate heterogeneity using a Dirichlet distribution, their testing is limited to a hyperparameter set to 0.9.

Therefore, in Section 4, we replicate the detection approach from Gupta et al. (2022) and further evaluate its effectiveness. We make the following three hypothesis:

- Does the historical gradients method outperform the one-round gradients method under heterogeneity with different levels of skewness?
- Does the historical gradients method outperform the one-round gradients method with varying numbers of participating clients?
- Does the historical gradients method outperform the one-round gradients method with different percentages of malicious clients?

Noting that, owing to the increased complexity in detecting label poisoning attacks as opposed to other types of attacks, our experiment is exclusively focused on label poisoning.

4 EXPERIMENTAL RESULTS

We train one CNN for MNIST classification using different numbers of clients (n=10,20,50), skewness factors (s=0.3,0.5,0.7) and malicious client rates (0.125,0.200,0.275,0.350,0.425). This CNN architecture consists of two 5x5 convolutional layers. The first convolutional layer has 10 output channels, followed by a 2x2 max-pooling operation and ReLU activation. The second convolutional layer has 20 output channels, utilizes a 5x5 kernel, applies 2D dropout, followed by another 2x2 max-pooling operation and ReLU activation. After the convolutional layers, there is a fully connected layer with 50 units, utilizing dropout regularization and ReLU activation. Finally, there is an output layer with 10 units, representing the number of classes in the classification task. The total number of parameters in the network is 21840. We use this CNN architecture to study how FedAvg and DP-SGD performs in non-i.i.d setting with different combinations of clients, skewness factors and malicious client rates. In training, we set training epochs to be 60. Also, we use Adam optimizer with learning rate of 0.001 and torch library's built-in cross entropy loss function.

Rate	Skewness Factor	Accuracy Historical Cluster	Accuracy Current Cluster
0.125	0.3	1	1
0.125	0.5	0.875	**0.95**
0.125	0.7	0.6333	**0.8**
0.2	0.3	1	1
0.2	0.5	1	1
0.2	0.7	0.6458	**0.825**
0.275	0.3	1	1
0.35	0.3	1	1
0.425	0.3	1	1
0.475	0.3	0.9833	0.9833

Table 1: Accuracy of identifying attacker (N = 20 clients) under FedAvg

Figure 2: Accuracy of identifying attacker (N = 20 clients) under FedAvg

Rate	Skewness Factor	Accuracy Historical Cluster	Accuracy Current Cluster
0.125	0.3	**0.9778**	0.8056
0.125	0.5	0.4167	**0.4944**
0.125	0.7	0.6694	**0.7361**
0.2	0.3	**0.9817**	0.9683
0.2	0.5	0.1783	**0.5517**
0.275	0.3	**0.9654**	0.9346
0.275	0.5	0.2295	**0.5359**
0.35	0.3	**0.9863**	0.9833
0.35	0.5	0.2118	**0.5118**
0.425	0.3	**0.9698**	0.9508
0.425	0.5	0.1857	**0.4730**
0.475	0.3	**0.9703**	0.9080
0.475	0.5	**0.3457**	0.2616
0.475	0.7	0.1889	**0.4796**

Table 2: Accuracy of identifying attacker (N = 50 clients) under FedAvg

Figure 3: Accuracy of identifying attacker (N = 50 clients) under FedAv

Rate	Skewness Factor	Accuracy Historical Cluster	Accuracy Current Cluster
0.125	0.3	1	1
0.125	0.5	0.875	**0.95**
0.125	0.7	0.6333	**0.8**
0.2	0.3	1	1
0.2	0.5	1	1
0.2	0.7	0.6458	**0.825**
0.275	0.3	1	1
0.275	0.5	0.9433	**0.9667**
0.275	0.7	0.8067	**0.8433**
0.35	0.3	1	1
0.35	0.5	**0.969**	0.9429
0.35	0.7	0.7714	**0.8071**
0.425	0.3	1	1
0.425	0.5	0.8354	**0.95**
0.425	0.7	0.9104	**0.9292**
0.475	0.3	0.9833	0.9833
0.475	0.5	0.8352	**0.9352**
0.475	0.7	0.1889	**0.4796**

Figure 4: Accuracy of identifying attacker (N = 20 clients) under DP-SGD

Rate	Skewness Factor	Accuracy Historical Cluster	Accuracy Current Cluster
0.125	0.3	**0.9778**	0.8056
0.125	0.5	0.4167	**0.4944**
0.125	0.7	0.6694	**0.7361**
0.2	0.3	**0.9817**	0.9683
0.2	0.5	0.1783	**0.5517**
0.275	0.3	**0.9654**	0.9346
0.275	0.5	0.2295	**0.5359**
0.35	0.3	**0.9863**	0.9833
0.35	0.5	0.2118	**0.5118**
0.425	0.3	**0.9698**	0.9508
0.425	0.5	0.1857	**0.4730**
0.475	0.3	**0.9703**	0.9080
0.475	0.5	**0.3457**	0.2616
0.475	0.7	0.1889	**0.4796**

Figure 5: Accuracy of identifying attacker (N = 50 clients) under DP-SGD

From figures above, we can see that performance is as good as or even better in some settings without long HoG. For **hypothesis 1**, after aggregating Fig. 2-5 by skewness factor, we see historical gradient method never surpasses current gradient method in average accuracy for FedAvg. It manages to beat current gradient only in DP-SGD with skewness=0.3 For **hypothesis 2**, after aggregating Fig. 2-5 by number of devices, we see historical gradient method never surpasses current gradient method in average accuracy for FedAvg and DP-SGD. For **hypothesis 3**, after aggregating Fig. 2-5 by rate, we see historical gradient method never surpasses current gradient method in average accuracy for FedAvg and DP-SGD.

5 Conclusion and Future Work

Our study suggests that under heterogeneous FL, using historical weights isn't always more effective than one-round weights, which is particularly evident in scenarios with huge number of clients and high heterogeneity. In these cases, one-round weights often perform better. For future research in less extreme settings, historical weights could be useful for classification, but in more severe cases, one-round weights are recommended for efficiency and accuracy.

Although our results typically favor one-round over historical weights, both this study and Gupta et al. (2022) have focused only on the 'mean' trajectory statistic. To thoroughly evaluate historical weights, more comprehensive statistics should be analyzed, similar to our approach with historical weights.

If historical weights are found to be very effective one day, we recommend their use in continuous FL contexts, which proposed by Wang et al. (2022). Here, attacker filtering occurs at each convergence of the global model, offering a more appropriate environment for historical weights, avoiding the high computational demands of using them in every round.

References

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2938–2948. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/bagdasaryan20a.html.

Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 634–643. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/bhagoji19a.html.

Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener (eds.), *Automata, Languages and Programming*, pp. 1–12, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-35908-1.

Pinar Erbil and M. Emre Gursoy. Detection and mitigation of targeted data poisoning attacks in federated learning. In *2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pp. 1–8, 2022. doi: 10.1109/DASC/PiCom/CBDCom/Cy55231.2022.9927914.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to Byzantine-Robust federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1605–1622. USENIX Association, August 2020. ISBN 978-1-939133-17-5. URL https://www.usenix.org/conference/usenixsecurity20/presentation/fang.

Ashish Gupta, Tie Luo, Mao V. Ngo, and Sajal K. Das. Long-short history of gradients is all you need: Detecting malicious and unreliable clients in federated learning. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian D. Jensen, and Weizhi Meng (eds.), *Computer Security – ESORICS 2022*, pp. 445–465, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-17143-7.

Li Huang, Andrew L. Shea, Huining Qian, Aditya Masurkar, Hao Deng, and Dianbo Liu. Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *Journal of Biomedical Informatics*, 99: 103291, 2019. ISSN 1532-0464. doi: https://doi.org/10.1016/j.jbi.2019.103291. URL https://www.sciencedirect.com/science/article/pii/S1532046419302102.

Najeeb Jebreel and Josep Domingo-Ferrer. Fl-defender: Combating targeted attacks in federated learning, 2022.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning, 2021.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020. doi: 10.1109/MSP.2020.2975749.

Fan Mo, Anastasia Borovykh, Mohammad Malekzadeh, Hamed Haddadi, and Soteris Demetriou. Quantifying information leakage from gradients, 05 2021.

Viraaji Mothukuri, Reza M. Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2021. ISSN 0167-739X. doi: https://doi.org/10.1016/j.future.2020.10.007. URL https://www.sciencedirect.com/science/article/pii/S0167739X20329848.

Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 739–753, 2019. doi: 10.1109/SP.2019.00065.

Solmaz Niknam, Harpreet S. Dhillon, and Jeffery H. Reed. Federated learning for wireless communications: Motivation, opportunities and challenges, 2020.

Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. 01 2021. doi: 10.14722/ndss.2021.24498.

Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems, 2020.

Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Demystifying membership inference attacks in machine learning as a service. *IEEE Transactions on Services Computing*, 14:2073–2089, 2019. URL `https://api.semanticscholar.org/CorpusID:86836429`.

Shuaiqi Wang, Jonathan Hayase, Giulia Fanti, and Sewoong Oh. Towards a defense against backdoor attacks in continual federated learning. *arXiv preprint arXiv:2205.11736*, 2022.

Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning, 2018.

A Appendix

From experiments, we consistently see that the global model in the setting with lowest number of devices (n=10) shows the best absolute performance and that with highest number of devices shows the worst absolute performance, in spite of different combinations of skewness factor and malicious rate, as one can see from comparing Fig. 6-11 in the appendix, for both FedAvg and DP-SGD. As expected, all settings suffer from model performance deterioration as the malicious rate increases. Generally, the performance of DP-SGD is slightly worse than FedAvg for all same combinations of setting, but that is to be expected due to the trade-off between performance and privacy. However, it is worth noting the stark visual contrast that one can easily see in Fig 11 from Fig 9. Although the global model with lowest malicious rate and lowest skewness factor performs the best, the experiment result not only shows more interpretable outcome, but also more variability in contrast to Fig 9.

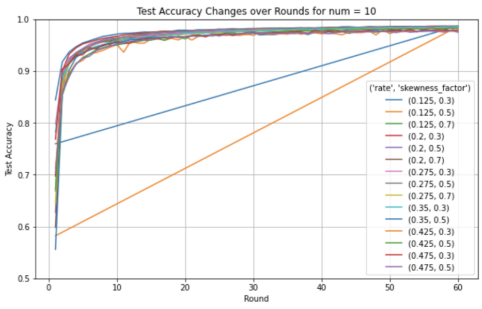

Figure 6: FedAvg: Test accuracy over 60 rounds for n=10

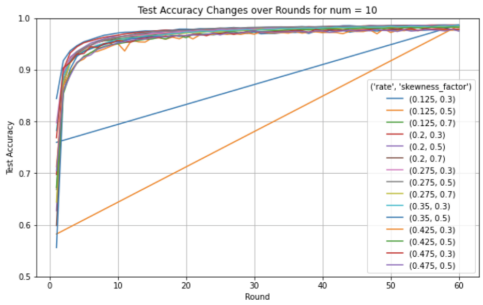

Figure 7: DP-SGD: Test accuracy over 60 rounds for n=10

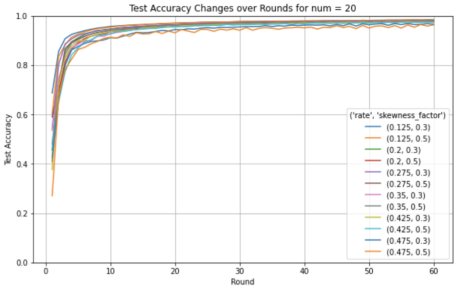

Figure 8: Test accuracy of FedAvg over 20 clients

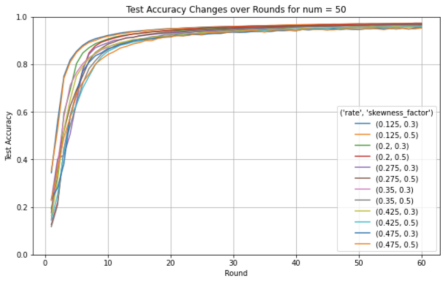

Figure 9: Test accuracy of FedAvg over 50 clients

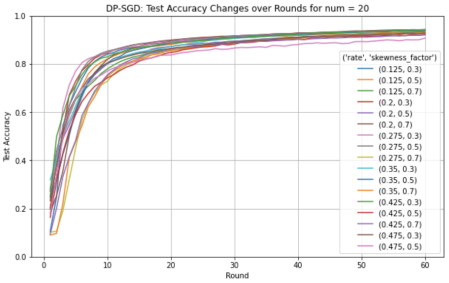

Figure 10: Test accuracy of DP-SGD over 20 clients

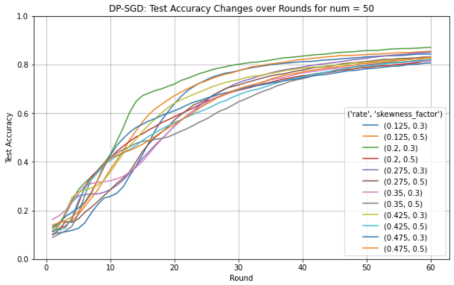

Figure 11: Test accuracy of DP-SGD over 50 clients